提出日：2024 年 5 月 10 日

先進情報プロジェクト実習

テーマ IT・1 第 3 回レポート

学籍番号：C0A22113
氏名：成田彩華

第１章　　ログインするプログラムを作成する

「/var/www/html」に「login」というディレクトリを作成し、その中に「login.php」を作成する。「login.php」の内容は以下の参考サイト「PHP ログイン機能サンプル」にある index.php を一部変更したものである。同じサイトの main.php も、一部変更して loginmain.php にする。なお、この二つのファイルは授業サイトからダウンロードできる。

以下が参考サイトの PHP ログイン機能サンプルである。

https://qiita.com/ShibuyaKosuke/items/83c21b6cc2dff2a93f72

参考サイトの index.php のログインユーザとパスワードをデータベース接続し、判定している部分を以下のようにコードで直接指定する形に変更する。また、ユーザとパスワードが一致した際 main.php に移動していたのを loginmain.php に移動するように変更する。

```
// エラーがない時
    if (count($err) === 0) {


        if(strcmp($user_name, "root") == 0 && strcmp($password, "abcd1234") == 0){
            $_SESSION['login_user'] = "root";
            header('Location:loginmain.php');
            return;
        }
        $err['login'] = ‘ログインに失敗しました。’;
    }
}
```

第２章　　参考サイトの PHP と MySQL システムを動かす

第 1 章の参考「PHP ログイン機能サンプル」にあるログインシステムを動かしてみる。「/var/www/html/login」にファイルを設置する。

第１節　MySQL をインストールする

MySQL をインストールする。以下のコマンドを入力し、インストールを進める。

```
$ sudo apt install mysql-server
この操作後に追加で 242 MB のディスク容量が消費されます。
続行しますか? [Y/n]
```

このように聞かれたら、Y を入力し、インストールを続行する。

第 2 節　MySQL のセキュリティを設定する

次に付属のセキュリティスクリプトを実行する。以下のコマンドを入力し、実行する。

```
$ sudo mysql_secure_installation
```

VALIDATE PASSWORD COMPONENT をインストールするか聞かれたら「Y」にする。

```
VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No: Y
```

パスワードの強度を聞かれたら最強強度の「2」にする。

```
Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG: 2
```

パスワードの強度を 2 にすると 8 文字以上、数字、大文字小文字混合、特殊記号、辞書ファイル検索がパスワードの条件となる。辞書ファイルの中の単語とパスワードが比較され、いずれかが一致するとパスワードが拒否されるようになる。

anonymous users は Remove するので「Y」にする。

```
By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : Y
```

root でリモートから login できないようにするかは「Y」にする。

```
Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : Y
```

test データベースを消すかは「Y」にする。

```
By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.

Remove test database and access to it? (Press y|Y for Yes, any other key for No) : Y
 - Dropping test database...
Success.

 - Removing privileges on test database...
Success.
```

privilege tables をリロードするかは「Y」にする。

```
Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : Y
Success.

All done!
```

MySQL のパスワードを設定していく。次のコマンドで MySQL に入る。

```
$ sudo mysql -u root
```

ALTER コマンドで次のようにパスワードを設定する。なお、今回の root ユーザーのパスワードは「passwordA1!」にしておく。

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password by 'passwordA1!';
```

「mysql> exit」で終了する。

再度、付属のセキュリティスクリプトを実行する。

```
$ sudo mysql_secure_installation
```

「Enter password for user root:」と聞かれたら、先ほどの「passwordA1!」を入力する。なお、パスワードの入力中は入力している文字が見えない。

root のパスワードを変更するか聞かれたら「No」を入力する。

```
Change the password for root ? ((Press y|Y for Yes, any other key for No) : No
```

この後 4 つほど聞かれるが、全て「Y」にする。

```
Remove anonymous users? (Press y|Y for Yes, any other key for No) : Y
```

```
Disallow root login remotely? (Press y|Y for Yes, any other key for No) : Y
Remove test database and access to it? (Press y|Y for Yes, any other key for No) : Y
Reload privilege tables now? (Press y|Y for Yes, any other key for No) : Y
```

「All done!」と表示されればインストールが完了となる。

第３節　MySQL を動かす

MySQL を動かすには、以下のコマンドを入力する。

```
$ sudo mysql -u root –p
```

パスワードを求められるため、先ほど設定した「passwordA1!」を入力する。MySQL を終了したい場合には、、「mysql> exit」で終了できる。

MySQL のデータベースにアクセスする、root 以外のユーザを作成しておく。以下のコマンドを入力してユーザを作成する。

```
mysql> create user ‘user1’@’localhost’identified by ‘passwordA1!’;
```

権限を与える。

```
mysql> GRANT CREATE, ALTER, DROP, INSERT, UPDATE, DELETE, SELECT, REFERENCES, RELOAD
on *.* TO 'user1'@'localhost' WITH GRANT OPTION;
```

サーバがキャッシュしたメモリを解放する。

```
mysql> FLUSH PRIVILEGES;
```

第４節　MySQL でデータベースを作成する

MySQL に入った後、以下のコマンドを入力し、データベースを作成する。

```
mysql> create database login_sample;
```

使用するデータベースを指定する。

```
mysql> use login_sample;
```

参考サイト「PHP ログイン機能サンプル」の内容を登録したのち、正しく登録されているか確認する。下の方にある次の内容である。

```
CREATE TABLE `User` (
  `id` INT(11) UNSIGNED NOT NULL AUTO_INCREMENT COMMENT 'AI',
  `user_name` VARCHAR(64) NOT NULL DEFAULT '' COMMENT '氏名',
  `password` VARCHAR(255) NOT NULL DEFAULT '' COMMENT 'パスワード',
  PRIMARY KEY (`id`),
```

```
  UNIQUE KEY `user_name` (`user_name`)
) ENGINE=INNODB DEFAULT CHARSET=utf8mb4;

INSERT INTO `User` (`id`, `user_name`, `password`) VALUES
(1,'user','$2y$10$ecRmAWY4n/jLa0tTzIaG7.SMhb1TfdROy3nXeG5aVZorUX1n6/WHO');
```

確認は次のように行う。

```
mysql> select * from User;
+----+-----------+--------------------------------------------------------------+
| id | user_name | password                                                     |
+----+-----------+--------------------------------------------------------------+
|  1 | user      | $2y$10$ecRmAWY4n/jLa0tTzIaG7.SMhb1TfdROy3nXeG5aVZorUX1n6/WHO |
+----+-----------+--------------------------------------------------------------+
```

参考サイト「PHP ログイン機能サンプル」のファイルを「var/www/html/login」に設置し、正しく動いているか確認する。

データベースにアクセスする際のユーザ名やパスワードを変更する必要があるので注意する。

```
    $dsn = 'mysql:host=localhost;dbname=login_sample;charset=utf8mb4;';
```

参考サイトの database.php の指定するデータベース名を sample から User に変え、上記の通りにする。

もし、エラーが出る場合、以下のコマンドを実行する。

```
$ sudo apt install php-mysql
この操作後に追加で 476 kB のディスク容量が消費されます。
続行しますか? [Y/n] Y
```

Apache2 のリロードをし、インストールした php-mysql を使えるようにする。

```
$ sudo systemctl reload apache2
```

以前作成した「info.php」で、PDO の「PDO drivers」に「mysql, sqlite」となっているか確認する。

第３章　　異なるユーザ名で同じパスワードのユーザを登録する

第１章、第２章の参考サイトにある「adduser.php」を使って新しいユーザの登録を行う。web ブラウザから adduser.php にアクセスし、ユーザ名は user2、パスワードは password を入力し、新規ユーザ登録の部分を押すことでユーザの登録を行う。MySQL からコマンドを使い、パスワードのハッシュティがどのように登録されているか確認する。

```
mysql> select * from User;
+----+-----------+--------------------------------------------------------------+
| id | user_name | password                                                     |
+----+-----------+--------------------------------------------------------------+
|  1 | user      | $2y$10$ecRmAWY4n/jLa0tTzIaG7.SMhb1TfdROy3nXeG5aVZorUX1n6/WHO |
|  2 | user2     | $2y$10$AFHFBDX6pVTdX2CZ6IO0IOR7/..Dp8S4rF4fnJZcXdgC.wqHIsHHm |
+----+-----------+--------------------------------------------------------------+
2 rows in set (0.00 sec)
```

追加したユーザを MySQL のコマンドを使って削除する。

```
mysql> delete from User where user_name = ‘user2’;
Query OK, 1 row affected(0.00 sec)
```

MySQL からコマンドを使い、ユーザが削除されていることを確認する。

```
mysql> select * from User;
+----+-----------+--------------------------------------------------------------+
| id | user_name | password                                                     |
+----+-----------+--------------------------------------------------------------+
|  1 | user      | $2y$10$ecRmAWY4n/jLa0tTzIaG7.SMhb1TfdROy3nXeG5aVZorUX1n6/WHO |
+----+-----------+--------------------------------------------------------------+
1 row in set (0.00 sec)
```

第４章　　パスワードハッシュ値を再現する

講義資料の hashsalt.php を使い、パスワードのハッシュを再現する。第 3 章の user2 のパスワードは次のようになっていた。

```
$2y$10$OjSlpj20AzKsMQm7kafZqOx/nLO.o5t/ADZ7ov/6giJWr7yafcMv2
```

このうち、$2y$の「2y」がアルゴリズムで、password_hash()関数で PASSWORD_BCRYPT を指定すると、CRYPT_BLOWFISH アルゴリズムが使用される。なお、PASSWORD_DEFAULT にしても bcrypt アルゴリズムが選択される。なお、10の「10」はコストである。2 の 10 乗で 1024 回ストレッチングされる。続く 22 文字が salt である。「OjSlpj20AzKsMQm7kafZqO」が salt ということになる。

ハッシュ値を再現するには crypt を使用して再現する。hashsalt.php を以下のように変更することでハッシュ値が適切に再現される。

```
<?php
echo crypt("passwordA1!", PASSWORD_BCRYPT, $options);
?>
```

第５章　　掲示板の表示機能を追加する

参考サイト「PHP ログイン機能サンプル」のファイルを「/var/www/html/bbs」に設置する。パーミッションはディレクトリの所有者に読み込み権限と実行権限を与える。また、所有者は www-data とする。

データベースを新たに作成し、指定する。

```
mysql> create database bbs_sample;
mysql> use bbs_sample;
```

テーブルの以下の内容を追加する。

```
CREATE TABLE `BBSlog` (
  `id` INT(11) UNSIGNED NOT NULL AUTO_INCREMENT COMMENT '番号',
  `article_title` VARCHAR(64) NOT NULL DEFAULT '' COMMENT 'タイトル',
  `user_name` VARCHAR(64) NOT NULL DEFAULT '' COMMENT '名前',
  `article_comments` VARCHAR(255) NOT NULL DEFAULT '' COMMENT '内容',
  PRIMARY KEY (`id`)
) ENGINE=INNODB DEFAULT CHARSET=utf8mb4;

INSERT INTO `BBSlog` (`id`, `article_title`, `user_name`, `article_comments`)
VALUES
```

```
    (1,'面白い','CSの人','この実験でよかった');
INSERT INTO `BBSlog` (`id`, `article_title`, `user_name`, `article_comments`)
VALUES
    (2,'はよ','Script Kiddy','ハッキングしたい。待ちきれない。');
INSERT INTO `BBSlog` (`id`, `article_title`, `user_name`, `article_comments`)
VALUES
    (3,'ためになりました','優等生','社会に出たときに役に立つ知識が身に付きました。');
```

登録された内容を確認する。

```
mysql> select * from BBSlog;
+----+--------------------------+-------------+-----------------------------------
---------------------------------+
| id | article_title            | user_name    |
article_comments                                                  |
+----+--------------------------+-------------+-----------------------------------
---------------------------------+
|  1 | 面白い                   | CSの人       | この実験でよかっ
た                                                  |
|  2 | はよ                     | Script Kiddy | ハッキングしたい。待ちきれない
。                                      |
|  3 | ためになりました         | 優等生       | 社会に出たときに役に立つ知識が身に付きました
。                        |
+----+--------------------------+-------------+-----------------------------------
---------------------------------+
3 rows in set (0.00 sec)
```

データベースは前の課題と被らないよう、「database.php」で以下のように指定して新しく作成する。

```
$dsn = 'mysql:host=localhost;dbname=bbs_sample;charset=utf8mb4;';
```

「index.php」と「main.php」に以下を追加もしくは書き換えし、ログインするとユーザ情報の代わりに掲示板の内容が表示されるようにする。

「index.php」に追加するのは以下の通りである。

```
$stmt2 = $pdo->prepare('SELECT * FROM BBSlog');
$stmt2->execute();
$rows2 = $stmt2->fetchAll();
```

```
$_SESSION['bbs'] = $rows2;
```

「main.php」に追加するのは以下の通りである。

```
$bbs_contents = $_SESSION['bbs'];
```

「main.php」の<body></body>の間を以下の通りに書き換える。

```
<?php foreach ($bbs_contents as $bbs_content) : ?>
    <?php foreach ($bbs_content as $key => $val) : ?>
        <p><?php echo h($key); ?> : <?php echo h($val); ?></p>
    <?php endforeach; ?>
<?php endforeach; ?>
```

第 6 章　　掲示板の表示を工夫する

以前扱った bbs.cgi と同じように、色や枠組みを使って PHP の掲示板が表示されるようにする。講義資料にあるサンプルコードを動作させて確認する。

PHP から HTML の色や枠組みを指定するには<style type=”text/css></style>の間で CSS で指定することができる。

```
<style type="text/css">
    <!--
        h1 { color: green }
        strong { color: blue; font-size: large }
        em { font-style: italic }
    -->
    </style>
```

データベースのデータを任意の位置に配置するには for 文でデータベースの列の要素文繰り返し、データを一つづつ取り出すことで任意の位置に配置している。

```
<?php foreach ($bbs_contents as $bbs_content) : ?>
        <div>
        <?php for ($i = 0; $i < count($bbs_content); $i++) : ?>
            <?php list($key, $var) = each($bbs_content); ?>
            <?php if ($i == 1) : ?>
                <strong><?php echo h($var); ?></strong><br>
            <?php endif; ?>
```

```
            <?php if ($i == 2) : ?>
                <em><?php echo h($var); ?></em><br><br>
            <?php endif; ?>
            <?php if ($i == 3) : ?>
                <?php echo h($var); ?>
            <?php endif; ?>
        <?php endfor; ?>
```

第７章　　掲示板に書き込めるようにする

ログインしたユーザだけが掲示板に書き込めるようにする。データベースは前の課題と被らないよう、「database2.php」を作成し、以下のように指定して新しく作成する。

```
$dsn = 'mysql:host=localhost;dbname=bbs_sample2;charset=utf8mb4;';
```

「index.php」と「main.php」を「index2.php」と「main2.php」として複写し、「database2.php」を参照するようにする。

先ほどと同じく、BBSlog のテーブルを作成するが、user_name にはログイン時に入力したアカウントの user_name が入るようにする。つまり、ユーザは書き込み時に表示される user_name を自由に変更できない。

グローバル変数として、次のように値がない状態の変数名を用意しておく必要がある。

```
require 'database2.php';
$login_user = [];
$bbs_contents = [];
$user_name = "";
```

「送信」ボタンが押されたら、次のように POST で値を渡せばよい。

```
// 「送信」ボタンが押されて、POST通信のとき
if (filter_input(INPUT_SERVER, 'REQUEST_METHOD') === 'POST') {
    $user_name = filter_input(INPUT_POST, 'user_name');
    $article_title = filter_input(INPUT_POST, 'article_title');
    $article_comments = filter_input(INPUT_POST, 'article_comments');

    if ($article_title === '') {
        $err['article_title'] = '題名は入力必須です。';
    }
    if ($article_comments === '') {
```

```
            $err['article_comments'] = '本文は入力必須です。';
        }
```

書き込みではなく、ログインした直後の場合には、次のようにすれば良い。なお、「$login_user」は配列であるため、「$user_name」の取り出し方に注意する。

```
    } else {
        $login_user = $_SESSION['login_user'];
        $bbs_contents = $_SESSION['bbs'];
        $user_name = $login_user['user_name'];
    }
```

題名や本文が書き込まれなかった場合には、次のようにエラーを表示して処理を中断する。

```
    <?php if (isset($err['article_title'])) : ?>
        <p class="error"><?php echo h($err['article_title']); ?></p>
    <?php endif; ?>
```

題名を入力する部分の HTML は以下のように作成した。

```
    <label for="article title">題名</label>
    <input name="article_title" type="text" size="60">
```

本文を入力する部分の HTML は以下のように作成した。

```
    <label for="article_comments">本文</label>
    <textarea name="article_comments" cols="60" rows="5"></textarea>
```

書き込み時に表示される user_name は、次のように hidden タイプの input タグで送信すれば良い。

```
    <input type="hidden" name="user_name" value="<?php echo $user_name; ?>"
```

以下に作成した「main2.php」のコードを示す。

```
    <?php
    ini_set('display_errors', true);
    error_reporting(E_ALL);
    /**
     * main2.php
     *
     * @since 2018/09/18
```

```
 */
session_start();

require 'database2.php';
//$id = 0;
$login_user = [];
$bbs_contents = [];
$user_name = "";
$err = [];

//$login_user = $_SESSION['login_user'];
//$bbs_contents = $_SESSION['bbs'];

//「送信」ボタンが押されて、POST通信のとき
if (filter_input(INPUT_SERVER, 'REQUEST_METHOD') === 'POST') {
    $user_name = filter_input(INPUT_POST, 'user_name');
    $article_title = filter_input(INPUT_POST, 'article_title');
    $article_comments = filter_input(INPUT_POST, 'article_comments');

    if ($article_title === '') {
        $err['article_title'] = '題名は入力必須です。';
        $login_user = $_SESSION['login_user'];
        $pdo = connect();
        $stmt2 = $pdo->prepare("SELECT * from BBSlog");
        $stmt2->execute();
        $rows2 = $stmt2->fetchAll();
            $bbs_contents = $rows2;
            $user_name = $login_user['user_name'];
    }

    if ($article_comments === '') {
        $err['article_comments'] = '本文は入力必須です。';
        $login_user = $_SESSION['login_user'];
        $pdo = connect();
        $stmt2 = $pdo->prepare("SELECT * from BBSlog");
        $stmt2->execute();
```

```
        $rows2 = $stmt2->fetchAll();
            $bbs_contents = $rows2;
            $user_name = $login_user['user_name'];
    }

    if (count($err) === 0) {

        //$id = $id + 1;

        $pdo = connect();

        $stmt = $pdo->prepare("INSERT INTO `BBSlog` (`id`, `article_title`,
`user_name`, `article_comments`) VALUES (null, ?, ?, ?);" );

        $params = [];
        //$params[] = (int)$id + 1;
        $params[] = $article_title;
        $params[] = $user_name;
        $params[] = $article_comments;
        $stmt->execute($params);
        $rows = $stmt->fetchAll();


        $stmt2 = $pdo->prepare("SELECT * from BBSlog");
        $stmt2->execute();
        $rows2 = $stmt2->fetchAll();

.           $bbs_contents = $rows2;

    }
} else {
    $login_user = $_SESSION['login_user'];
    $bbs_contents = $_SESSION['bbs'];
    $user_name = $login_user['user_name'];

}
```

```
?>
<!DOCTYPE HTML>
<html lang="ja">
    <head>
        <meta charset="UTF-8">
        <title>掲示板サンプル</title>
        <style type="text/css">
        <!--
            h1 { color: green }
            strong { color: blue; font-size: large }
            em { font-style: italic }
        -->
        </style>
    </head>
    <body>
        <h1>私の掲示板</h1>
        <p>ご自由に書き込んでください。</p>
        <form acrion="" method="post">
        <div>
            <style type"text/css">
                    .error {
                            color: red;
                    }
            </style>
            <?php if (isset($err['article_title'])) : ?>
            <p class="error"><?php echo h($err['article_title']); ?></p>
            <?php endif; ?>
            <label for="article title">題名</label>
            <input name="article_title" type="text" size="60"><br>

            <?php if (isset($err['article_comments'])) : ?>
            <p class="error"><?php echo h($err['article_comments']); ?></p>
            <?php endif; ?>

            <label for="article_comments">本文</label>
```

```
            <textarea name="article_comments" cols="60" rows="5"></textarea><br>
            <input type="hidden" name="user_name" value="<?php echo $user_name; ?>"/>

            <button type="submit">送信</button>
            <button type="reset">リセット</button>
        </form>
        </div>

        <hr>
        <?php foreach ($bbs_contents as $bbs_content) : ?>
            <?php for ($i = 0; $i < count($bbs_content); $i++) : ?>

                <?php $current = current($bbs_content); ?>
                <?php next($bbs_content); ?>
                <?php $key = key($bbs_content); ?>
                <?php $var = $current; ?>
                <?php if ($i == 1) : ?>
                    <strong><?php echo h($var); ?></strong><br>
                <?php endif; ?>
                <?php if ($i == 2) : ?>
                    <em><?php echo h($var); ?></em><br><br>
                <?php endif; ?>
                <?php if ($i == 3) : ?>
                    <?php echo h($var); ?>
                <?php endif; ?>
            <?php endfor; ?>
            </div><hr>
        <?php endforeach; ?>
    </body>
</html>
```

第８章　　表示されるユーザ名をハッシュ値にしてプライバシを保護する

掲示板の書き込みに表示されるユーザ名が、ログイン時のアカウント名と同様であると、攻撃者にアカウント名を教えることになり、プライバシ漏洩の危険性がある。例えば、ログイン時のアカウント名が「taro_kouka」で、書き込み時の名前も「taro_kouka」であるとすると、攻撃者はログイン時のアカウント名から「工科太郎」がユーザなのではないかと推測できる。そこで、表示されるユーザ名をハッシュ値にする。ソルトは「ABCDEFGHIJKLMNOPQRSTUO」の 22 文字に固定する。

データベースは前の課題と被らないよう、「database3.php」を作成し、以下のように指定して新しく作成する。

```
$dsn = 'mysql:host=localhost;dbname=bbs_sample3;charset=utf8mb4;';
```

「index2.php」と「main2.php」を「index3.php」と「main3.php」として複写し、「database3.php」を参照するようにする。

先ほどと同じく、BBSlog のテーブルを作成するが、user_name にはログイン時に入力したアカウントの user_name のハッシュ値が入るようにする。つまり、ユーザは書き込み時に表示される user_name を自由に変更できない。

「main2.php」のユーザ名を表示させる部分を以下のように書き換え、ハッシュ値を表示させる crypt 関数に渡してユーザ名のハッシュ値を表示させる。

```
<?php echo h(crypt($var, '$2y$10$ABCDEFGHIJKLMNOPQRSTUO')); ?>
```

この変更と database3.php の指定部分以外は「main2.php」と同じである。

第９章　　表示されるユーザ名のハッシュ値に salt をつける

ハッシュ値のユーザ名が表示されていても、ソルトが固定されて見えているため、攻撃者がログイン時のアカウント名を類推し、ハッシュ値が一致するか確認できてしまう。

「hashsalt.php」を複写した「hashsalt2.php」を「/var/www/html/login」ディレクトリに作成し、掲示板に表示されているハッシュ値から攻撃者がログイン時のアカウント名を類推する方法を試す。

まず、次の内容が掲示板に表示されているものとする。

```
$2y$10$ABCDEFGHIJKLMNOPQRSTUO7/pkMzWkveZxli.RHeEAS56jPeeUNn6
```

「hashsalt2.php」を適切に書き換え、これと同じハッシュ値が画面に表示されるようにする。ユーザ名が taro_kouka だと類推し、ハッシュ値を再現する。以下に「hashsalt2.php」を示す。

```
<?php
echo crypt('taro_kouka', '$2y$10$ABCDEFGHIJKLMNOPQRSTUO')
?>
```

salt を「ABCDEFGHIJKLMNOPQRSTUV」の 22 文字に固定すると「ABCDEFGHIJKLMNOPQRSTUO」で指定した場合と変わらず、

```
$2y$10$ABCDEFGHIJKLMNOPQRSTUO7/pkMzWkveZxli.RHeEAS56jPeeUNn6
```

になることが確認できる。